

Statistically optimal observables for global SMEFT fits

LHC EFT WG (Area 3)
24/10/23



VRIJE
UNIVERSITEIT
AMSTERDAM



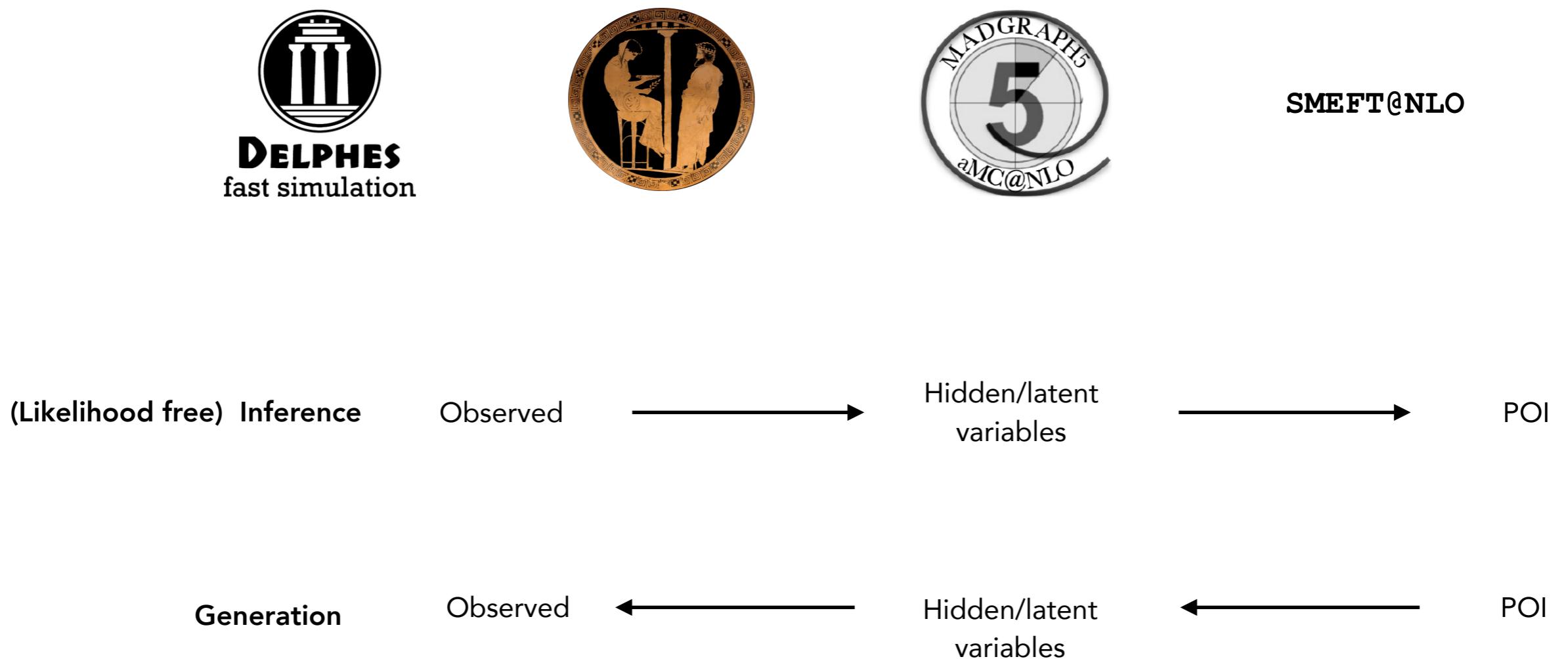
Jaco ter Hoeve
VU Amsterdam & Nikhef theory group

in collaboration with R. G. Ambrosio,
M. Madigan, J. Rojo and V. Sanz

Where theory and experiment meet

We are progressively moving through the simulation chain (latent space)

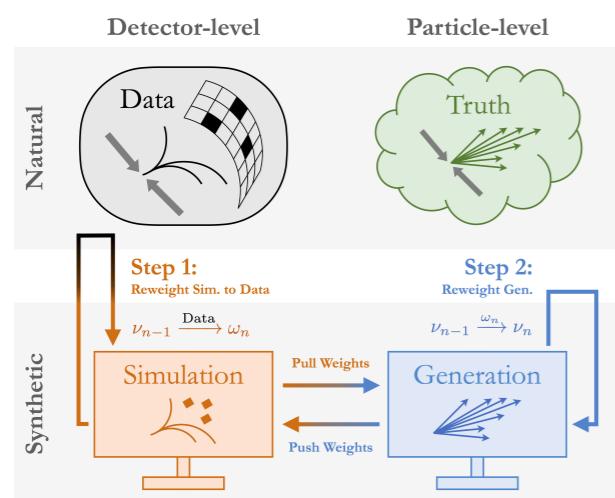
$$p(x|c) \sim \int dz_{\text{det}} dz_{\text{shower}} dz_{\text{parton}} p(x|z_{\text{det}}) p(z_{\text{det}}|z_{\text{shower}}) p(z_{\text{shower}}|z_{\text{parton}}) p(z_{\text{parton}}|c)$$



Where theory and experiment meet

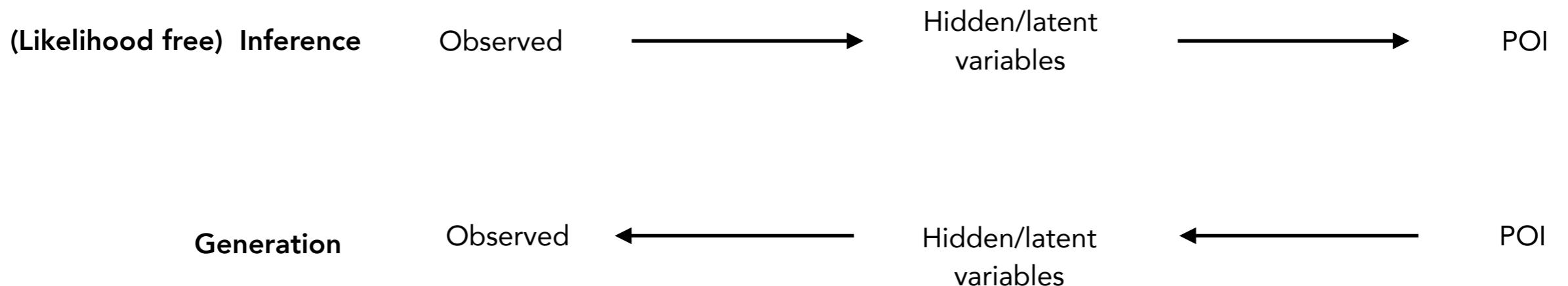
We are progressively moving through the simulation chain (latent space)

$$p(x|c) \sim \int dz_{\text{det}} dz_{\text{shower}} dz_{\text{parton}} p(x|z_{\text{det}}) p(z_{\text{det}}|z_{\text{shower}}) p(z_{\text{shower}}|z_{\text{parton}}) p(z_{\text{parton}}|c)$$



SMEFT@NLO

Unbinned unfolding, Omnidisk [1911.09107]



Likelihood free inference

- Starting from two balanced datasets \mathcal{D}_{SM} and \mathcal{D}_{EFT} drawn from $f(\mathbf{x} | \text{SM})$ and $f(\mathbf{x} | \text{EFT})$, we minimise e.g. the cross-entropy loss

$$L[g(\mathbf{x})] = -\frac{1}{N} \sum_{e \in \mathcal{D}_{\text{EFT}}} w_e \log(1 - g(\mathbf{x}_e)) - \frac{1}{N} \sum_{\mathcal{D}_{\text{SM}}} w_e \log g(\mathbf{x}_e)$$

Event weights

$\{m_{t\bar{t}}, \eta_l, \Delta\phi, \dots\}$

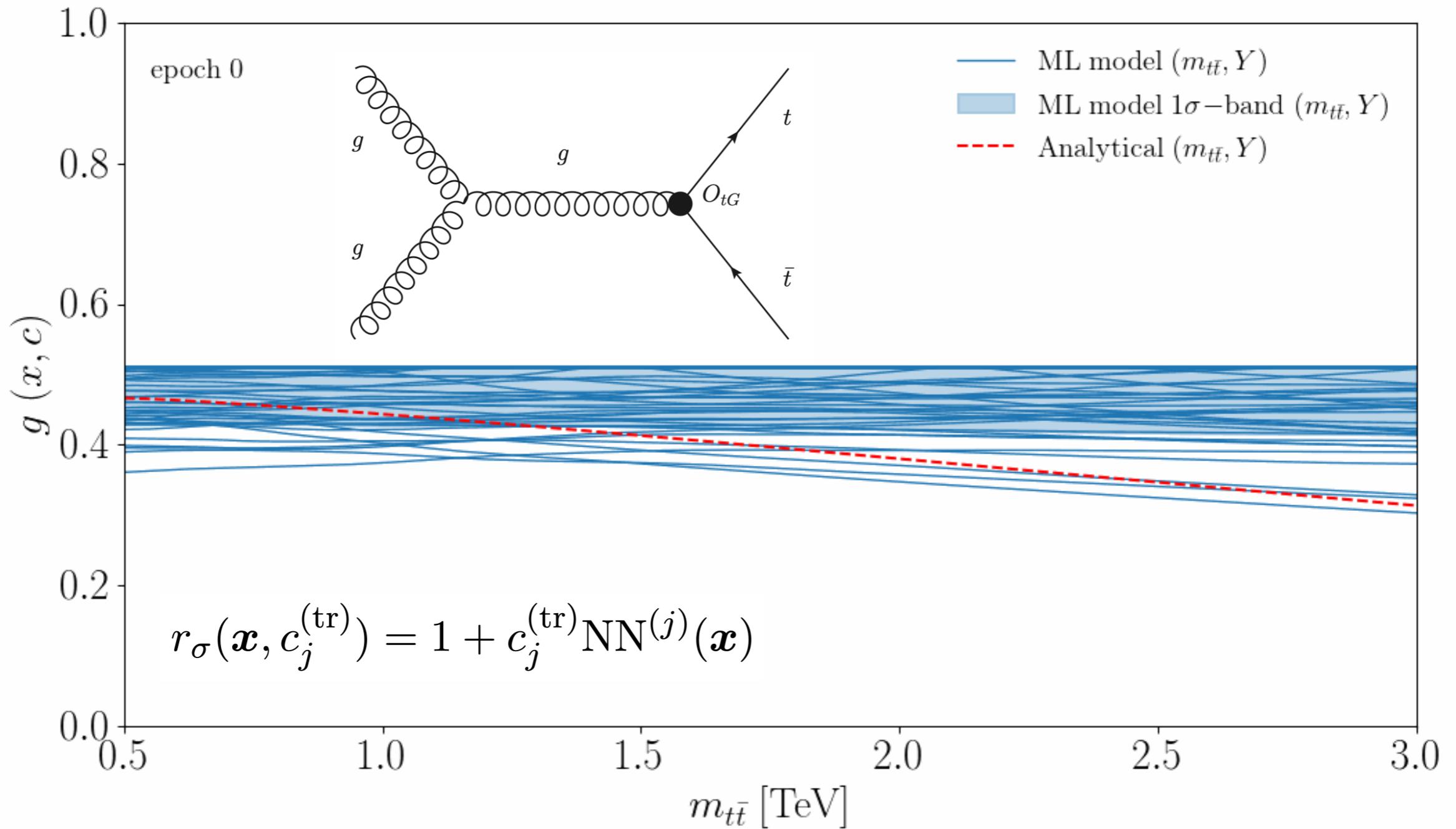
- The learned decision boundary $g(\mathbf{x})$ is one-to-one with the likelihood ratio (LR) as $N \rightarrow \infty$

$$\frac{\delta L}{\delta g} = 0 \implies \hat{g}(\mathbf{x}) = \left(1 + \frac{f(\mathbf{x} | \text{EFT})}{f(\mathbf{x} | \text{SM})} \right)^{-1} \equiv \frac{1}{1 + r(\mathbf{x})}$$

Parameterise with NNs

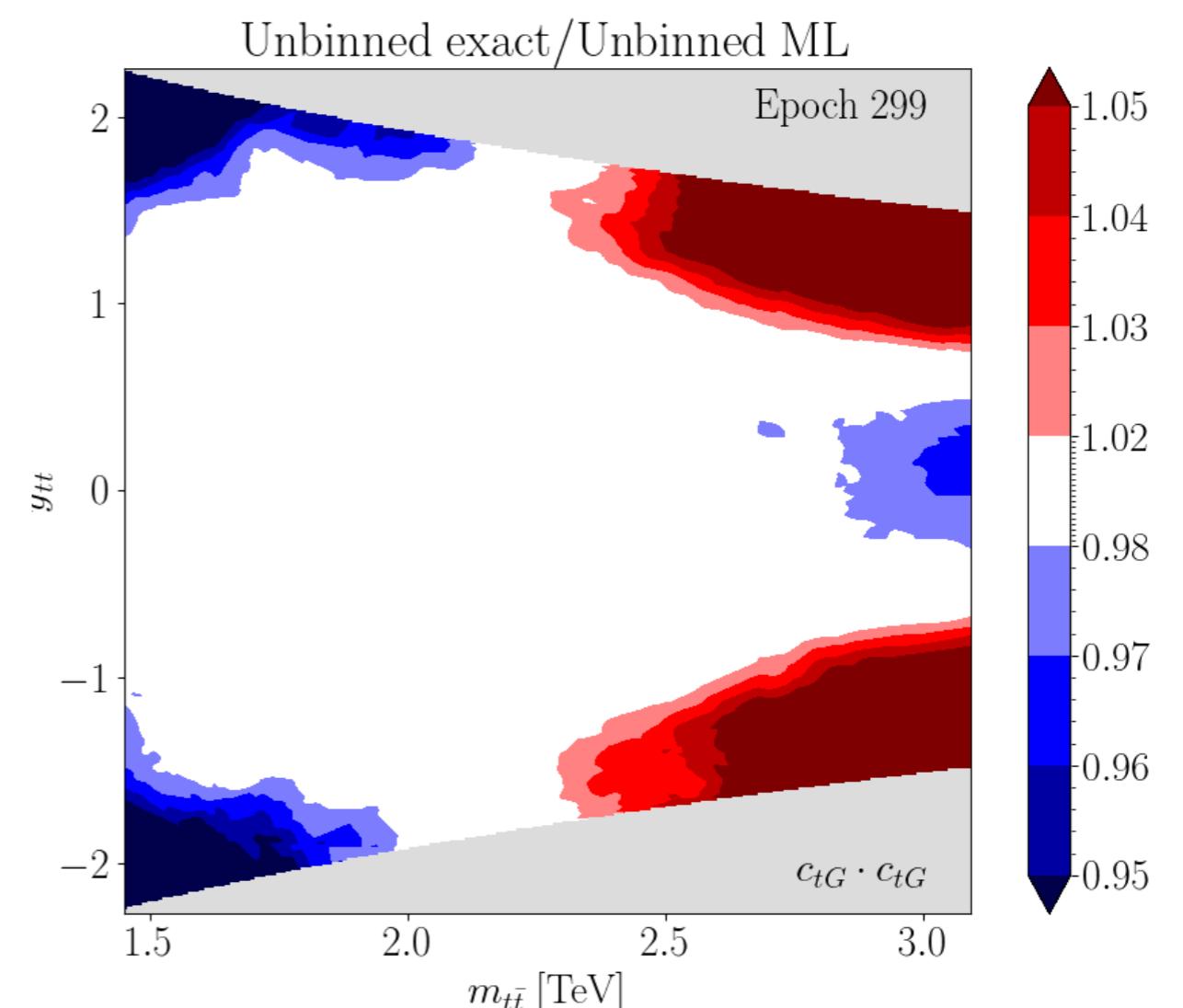
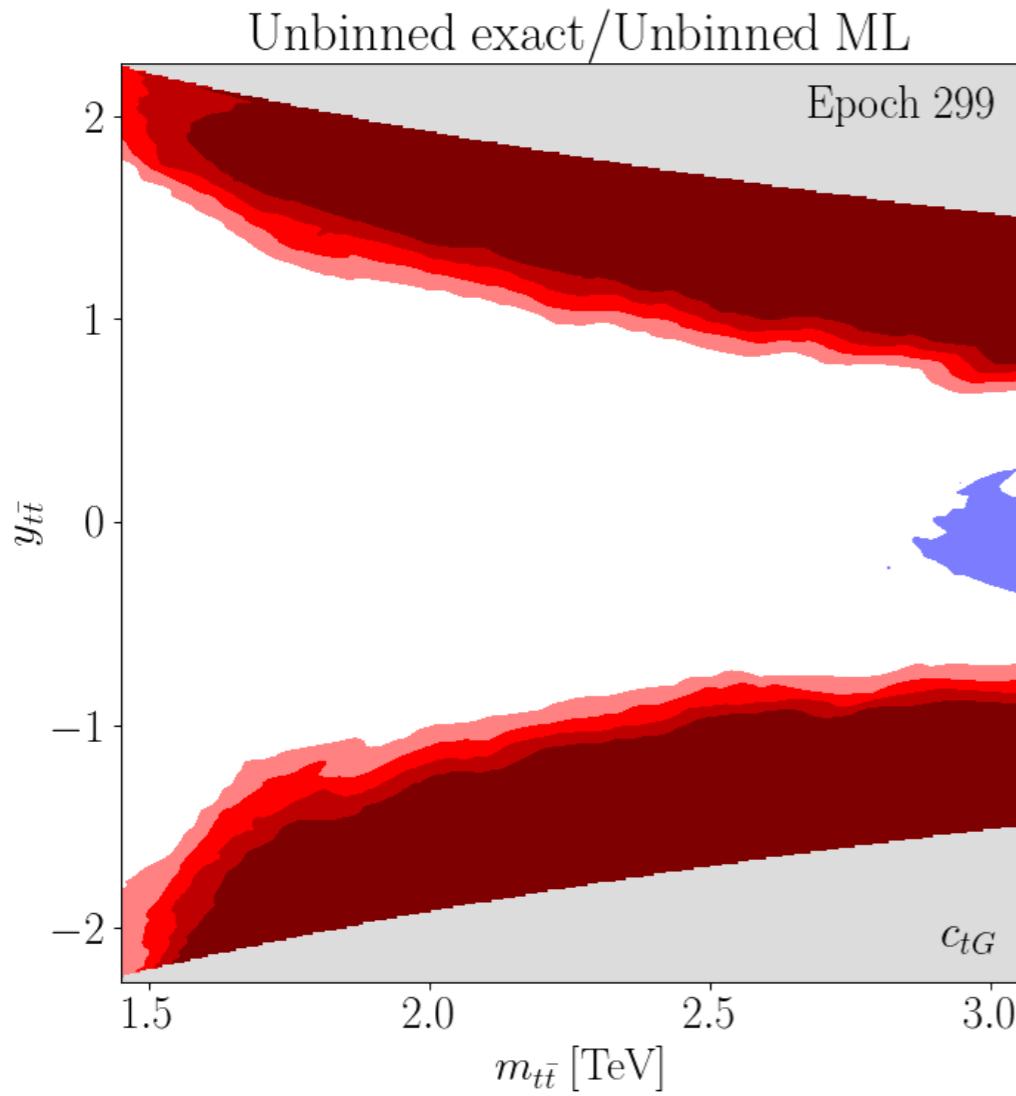
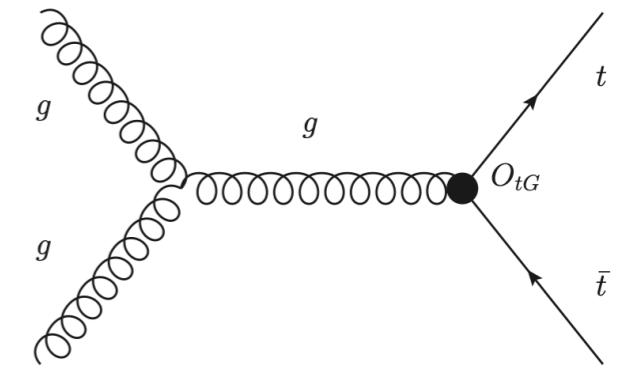
Likelihood free inference

$$L[g(\mathbf{x}, \mathbf{c})] = -\frac{1}{N} \sum_{e \in \mathcal{D}_{\text{eft}}} w_e \log(1 - g(\mathbf{x}_e, \mathbf{c})) - \frac{1}{N} \sum_{e \in \mathcal{D}_{\text{sm}}} w_e \log g(\mathbf{x}_e, \mathbf{c})$$



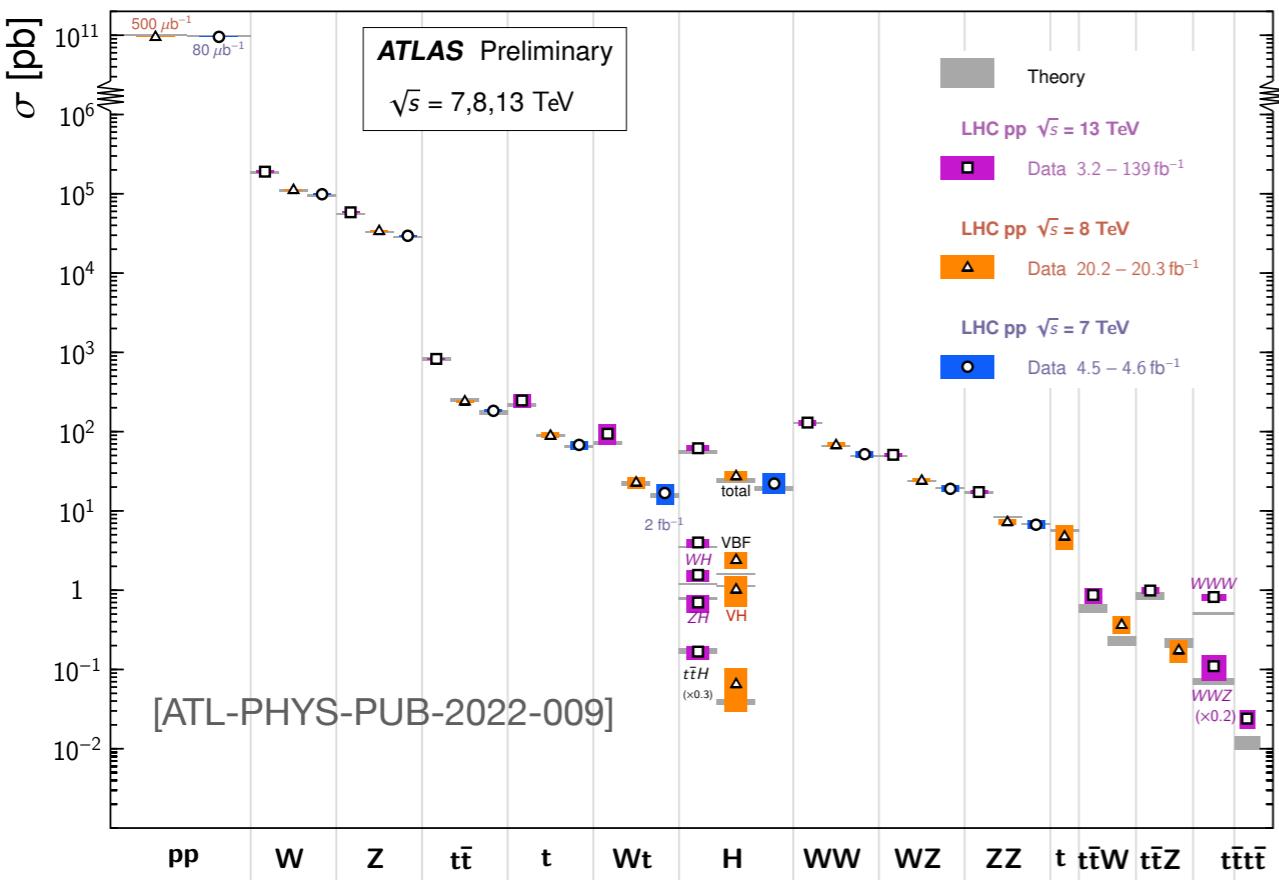
Likelihood free inference

2 kinematic features



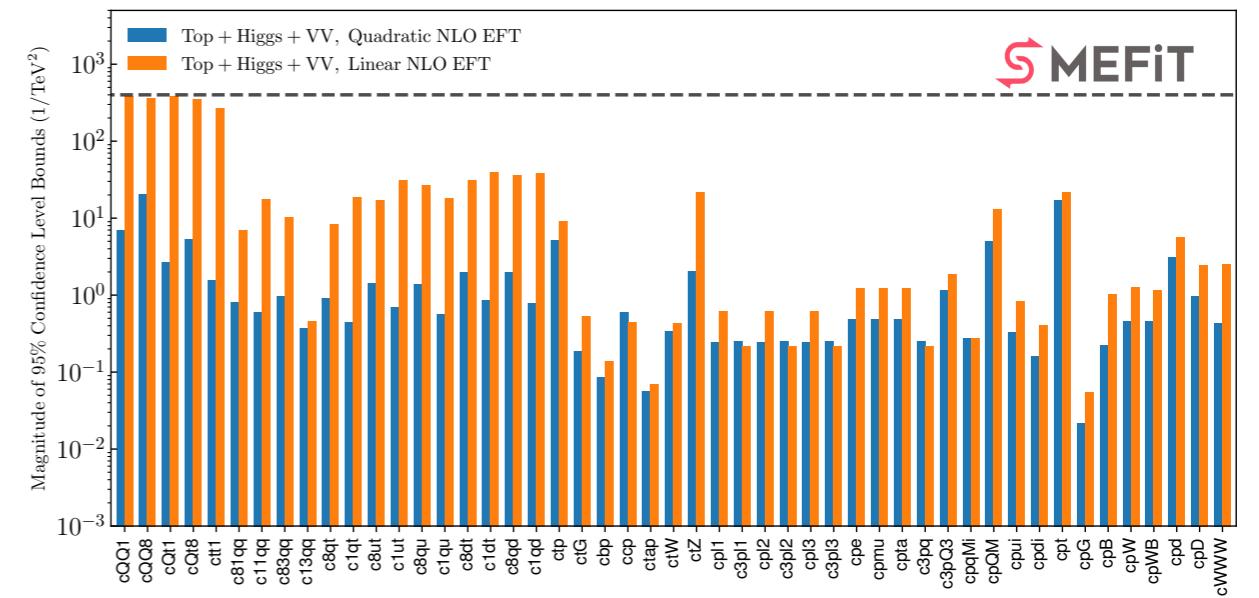
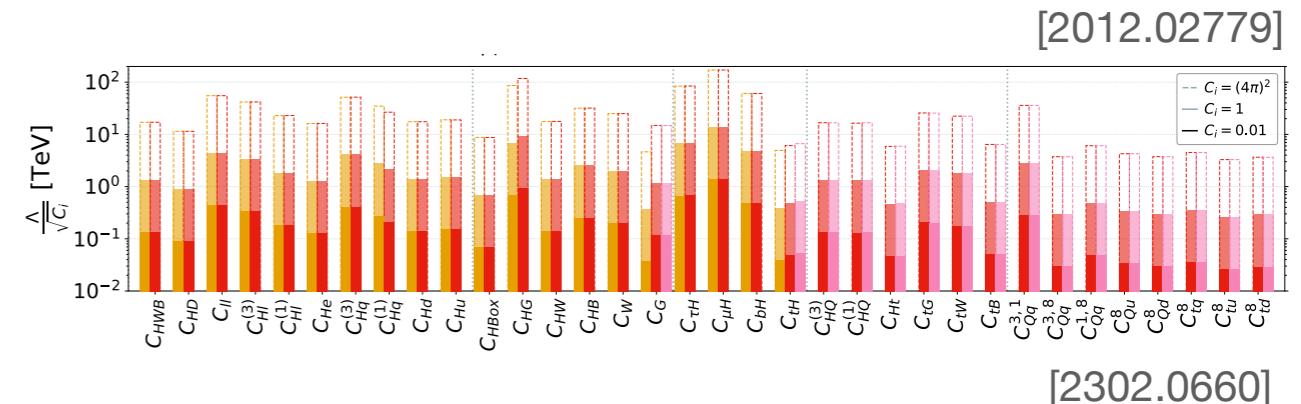
Why interesting for the SMEFT?

Standard Model Total Production Cross Section Measurements



$$\sigma(c) = \sigma_{\text{SM}} \left(1 + \sum_i N_{d6} \kappa_i c_i + \sum_{i < j} \tilde{\kappa}_{ij} c_i \cdot c_j \right)$$

$$\chi^2 \sim (\sigma_i(c) - \sigma_{i,\text{exp}}) (\text{cov}^{-1})_{ij} (\sigma_j(c) - \sigma_{j,\text{exp}})$$

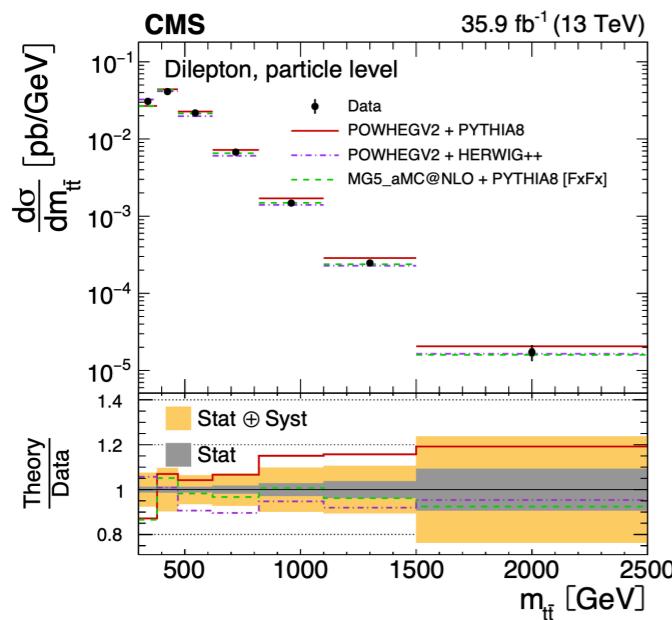


Why interesting for the SMEFT?

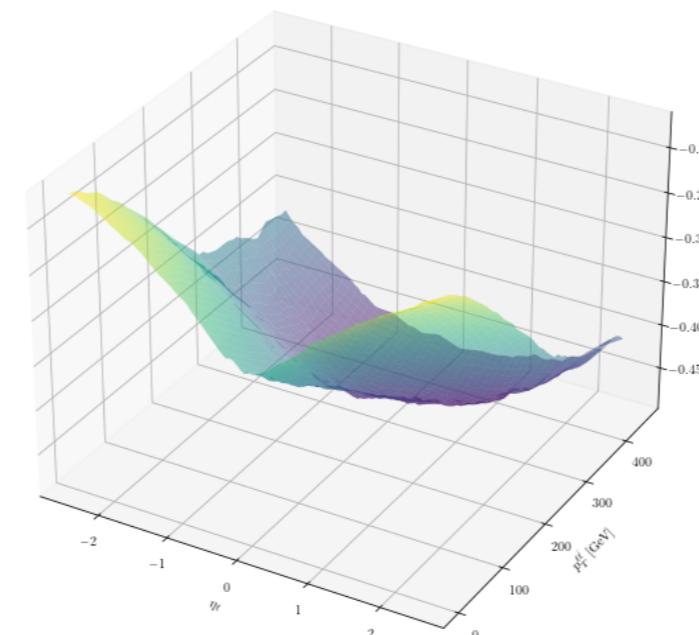
Standard Model Total Production Cross Section Measurements



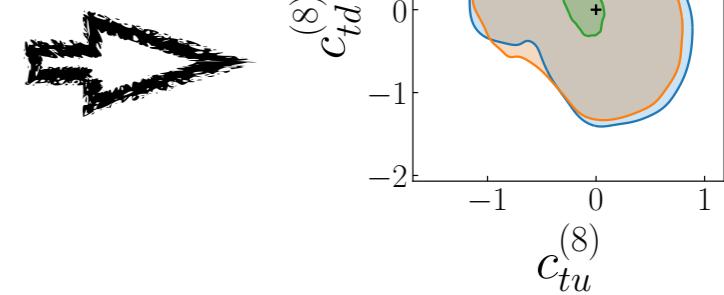
$$\chi^2 \sim (\sigma_i(c) - \sigma_{i,\text{exp}}) (\text{cov}^{-1})_{ii} (\sigma_j(c) - \sigma_{j,\text{exp}})$$



Binned, univariate

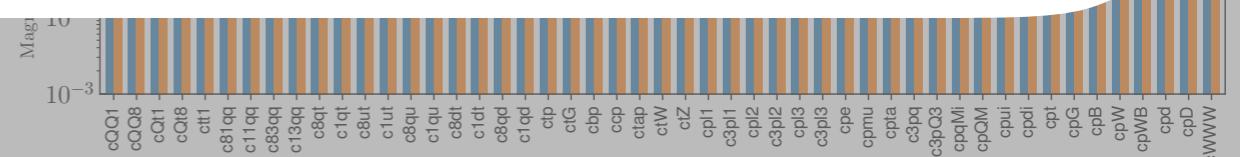


Unbinned, multivariate



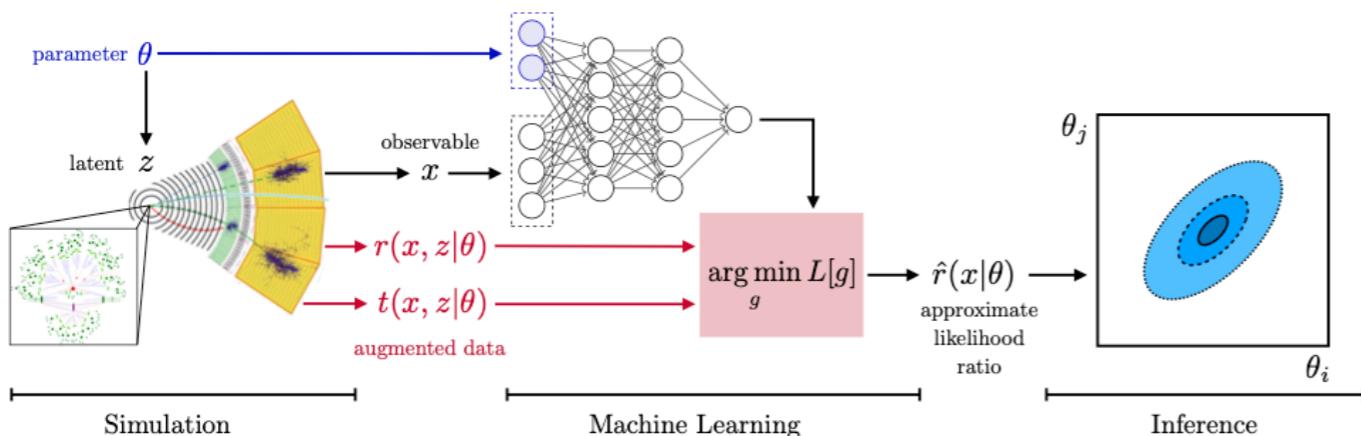
Optimal inference

$$\sigma(c) = \sigma_{\text{SM}} \left(1 + \sum_i \kappa_i c_i + \sum_{i < j} \kappa_{ij} c_i \cdot c_j \right)$$

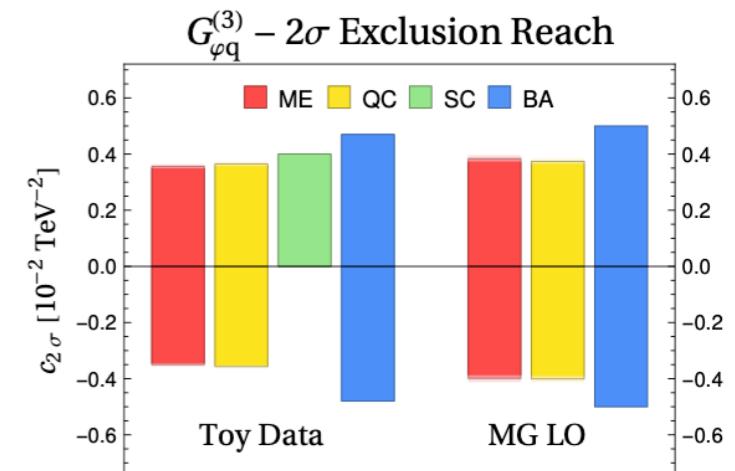


Related work

- ▶ Madminer series (J.Brehmer, K.Cranmer, G.Loupe et al.) [1907.10621, 1805.00020, ...]
- ▶ Parameterized classifiers for SMEFT (A. Glioti et al.) [2007.10356]
- ▶ Learning the EFT likelihood with tree boosting (R. Schöfbeck et al) [2205.12976]
- ▶ Back to the Formula (A. Butter, T. Plehn et al) [2109.10414]
- ▶ Boosted likelihood learning with event reweighting (A. Glioti et al) [2308.05704] See next talk!
- ▶ Designing Observables for Measurements with Deep Learning (O.Long, B. Nachman) [2310.08717]



[2010.06439]



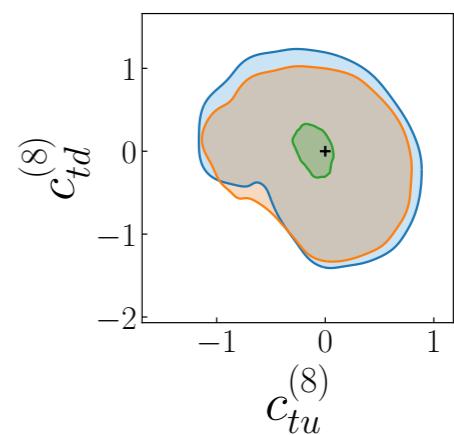
[2007.10356]

This talk

- Global efforts **reinterpret** existing “SM measurements” in an EFT context
- But which measurements are the most **sensitive** to EFT parameters?
 - Inclusive, single to multi-differential (which variables)
 - Binned or unbinned, which binning?

Framework needed to integrate unbinned multivariate observables into **global SMEFT fits**

- **Optimal bounds** on the EFT parameters
- Useful **diagnosis tool** to assess information loss



The ML4EFT framework

pip install ml4eft

<https://lhcfitnikhef.github.io/ML4EFT>

2211.02058 R. Gomez Ambrosio, JtH, M. Madigan, J. Rojo, V.Sanz

Open-source NN-based python framework for the integration of unbinned multivariate observables into global SMEFT fits

- ▶ **Goal:** provide optimal constraints on the SMEFT
- ▶ **Diagnostic tool:** what is the information loss incurred by a particular choice of bins?
- ▶ **Projections:** how will SMEFT constraints improve if unbinned data are made available?

The screenshot shows a Python documentation page for the `ml4eft.core.classifier.Fitter` class. The left sidebar contains a navigation tree for the ML4EFT documentation, including sections for CODE (Installation, Tutorial), ml4eft (ml4eft.analyse, ml4eft.core), ml4eft.core.classifier (ml4eft.core.classifier.Classifier, ml4eft.core.classifier.ConstraintA, ml4eft.core.classifier.CustomActiv, ml4eft.core.classifier.EventDatabase, ml4eft.core.classifier.Fitter, ml4eft.core.classifier.MLP, ml4eft.core.classifier.PreProcessin, ml4eft.core.th_predictions, ml4eft.core.truth, ml4eft.limits, ml4eft.plotting, ml4eft.preproc), RESULTS (Unbinned multivariate observables for global SMEFT analyses from machine learning), and BIBLIOGRAPHY (Bibliography). The right panel details the `Fitter` class, showing its definition, constructor, parameters, methods, and source code links.

```
class ml4eft.core.classifier.Fitter(json_path, mc_run, c_name, output_dir, print_log=False)
    [source]
Bases: object
Training class
__init__(json_path, mc_run, c_name, output_dir, print_log=False) [source]
Fitter constructor
Parameters:
    • json_path (str) – Path to json run card
    • mc_run (int) – Replica number
    • c_name (str) – EFT coefficient for which to learn the ratio function
    • output_dir (str) – Path to where the models should be stored
    • print_log (bool, optional) – Set to true to print training progress to stdout, otherwise it prints to a log file only
Methods
__init__(json_path, mc_run, c_name, output_dir) Fitter constructor
load_data() Constructs training and validation sets
loss_fn(outputs, labels, w_e) Loss function
train_classifier(data_train, data_val) Starts the training of the binary classifier
training_loop(optimizer, train_loader, ...) Optimizes the classifier with optimizer on the training data set train_loader.
weight_reset(m) Resets the weights and biases associated with the model m.
```

Modular structure, easy to maintain, well documented .

Anticipating global fits

- Global EFT fits typically feature ~ 50 WCs and thus efficient scaling with the number of WCs becomes essential
- Solution: learn the coefficient functions separately and combine afterwards

$$r(\mathbf{x}, \mathbf{c}) \equiv \frac{d\sigma(\mathbf{x}, \mathbf{c})}{d\sigma(\mathbf{x}, \mathbf{0})} = 1 + \sum_{j=1}^{n_{\text{eft}}} r^{(j)}(\mathbf{x}) c_j + \sum_{j=1}^{n_{\text{eft}}} \sum_{k \geq j}^{n_{\text{eft}}} r^{(j,k)}(\mathbf{x}) c_j c_k$$

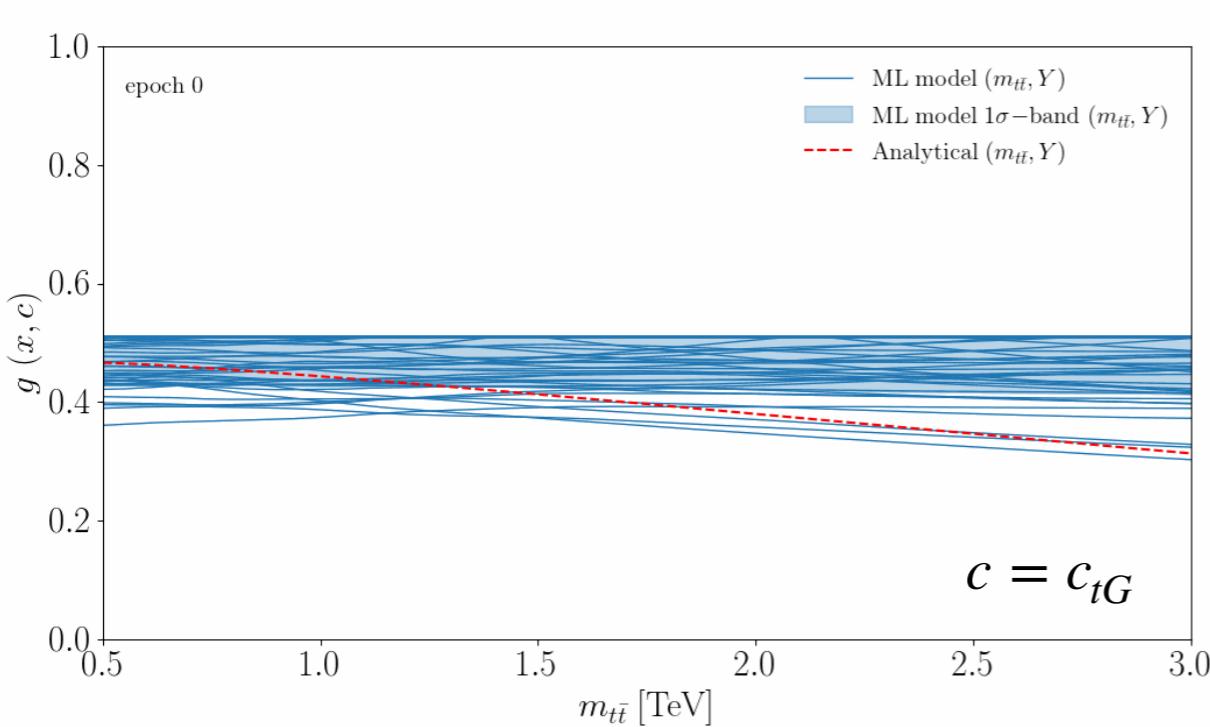
Example: to learn a single $r^{(j)}$, generate \mathcal{D}_{sm} and \mathcal{D}_{eft} at c_j up to $\mathcal{O}(\Lambda^{-2})$.
Then $r(\mathbf{x}, \mathbf{c}) = 1 + r^{(j)}(\mathbf{x}) c_j^{(\text{tr})}$ and training means

$$g(\mathbf{x}, c_j^{(\text{tr})}) = \left(1 + \left[1 + c_j^{(\text{tr})} \cdot \text{NN}^{(j)}(\mathbf{x}) \right] \right)^{-1} \quad \text{NN}^{(j)}(\mathbf{x}) \rightarrow r^{(j)}(\mathbf{x})$$

Uncertainty treatment

- Stick to a regime in which statistical uncertainties dominate over systematics
- Finite training data makes one subject to methodological uncertainties
- Solution: propagate uncertainties to the space of models by training multiple replicas

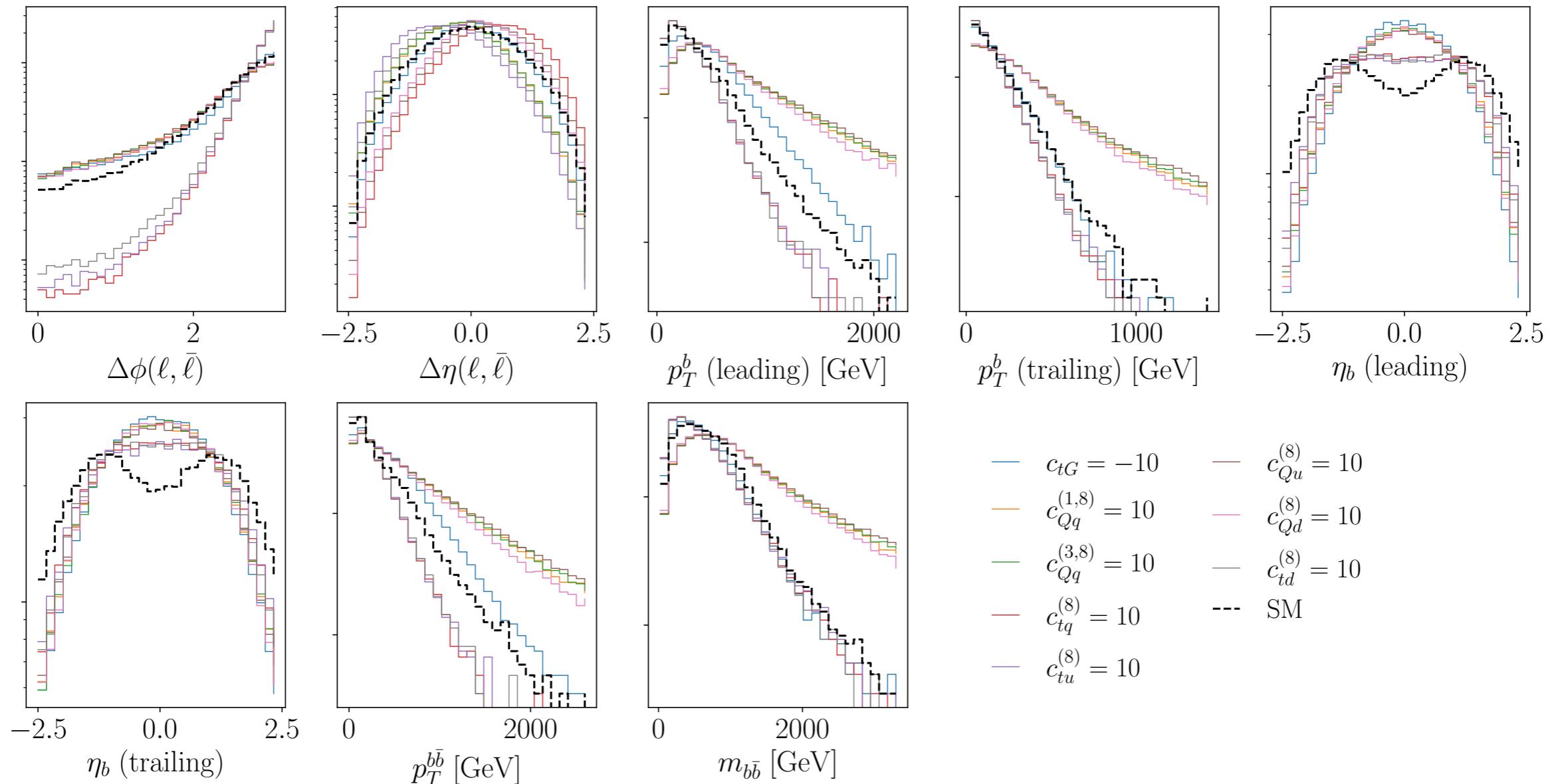
$$\hat{r}^{(i)}(\mathbf{x}, \mathbf{c}) \equiv 1 + \sum_{j=1}^{n_{\text{eft}}} \text{NN}_{\mathbf{i}}^{(j)}(\mathbf{x}) c_j + \sum_{j=1}^{n_{\text{eft}}} \sum_{k \geq j} \text{NN}_{\mathbf{i}}^{(j,k)}(\mathbf{x}) c_j c_k, \quad i = 1, \dots, N_{\text{rep}}$$



Process	N_{rep}	\tilde{N}_{ev} (per replica)	N_{nn}	#trainings
$pp \rightarrow t\bar{t}$	50	10^5	4	200
$pp \rightarrow t\bar{t} \rightarrow b\bar{b}\ell^+\ell^-\nu_\ell\bar{\nu}_\ell$	25	10^5	40	1000

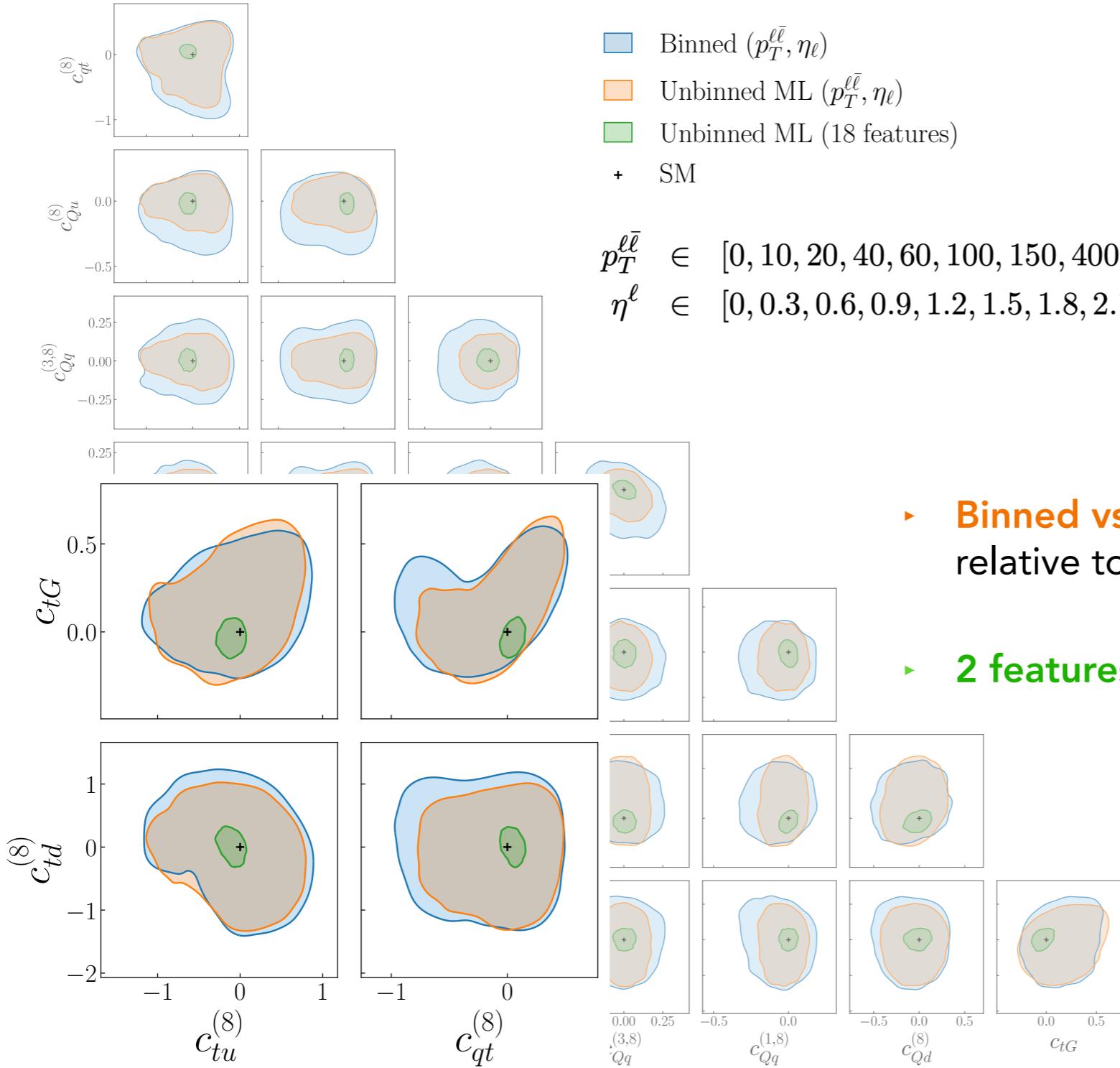
Let's go multivariate

- $pp \rightarrow t\bar{t} \rightarrow b\bar{b}\ell^+\ell^-\nu_\ell\bar{\nu}_\ell$: 18 features, 8 EFT coefficients
- $pp \rightarrow hZ \rightarrow b\bar{b}\ell^+\ell^-$: 7 features, 7 EFT coefficients



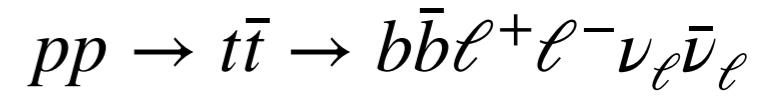
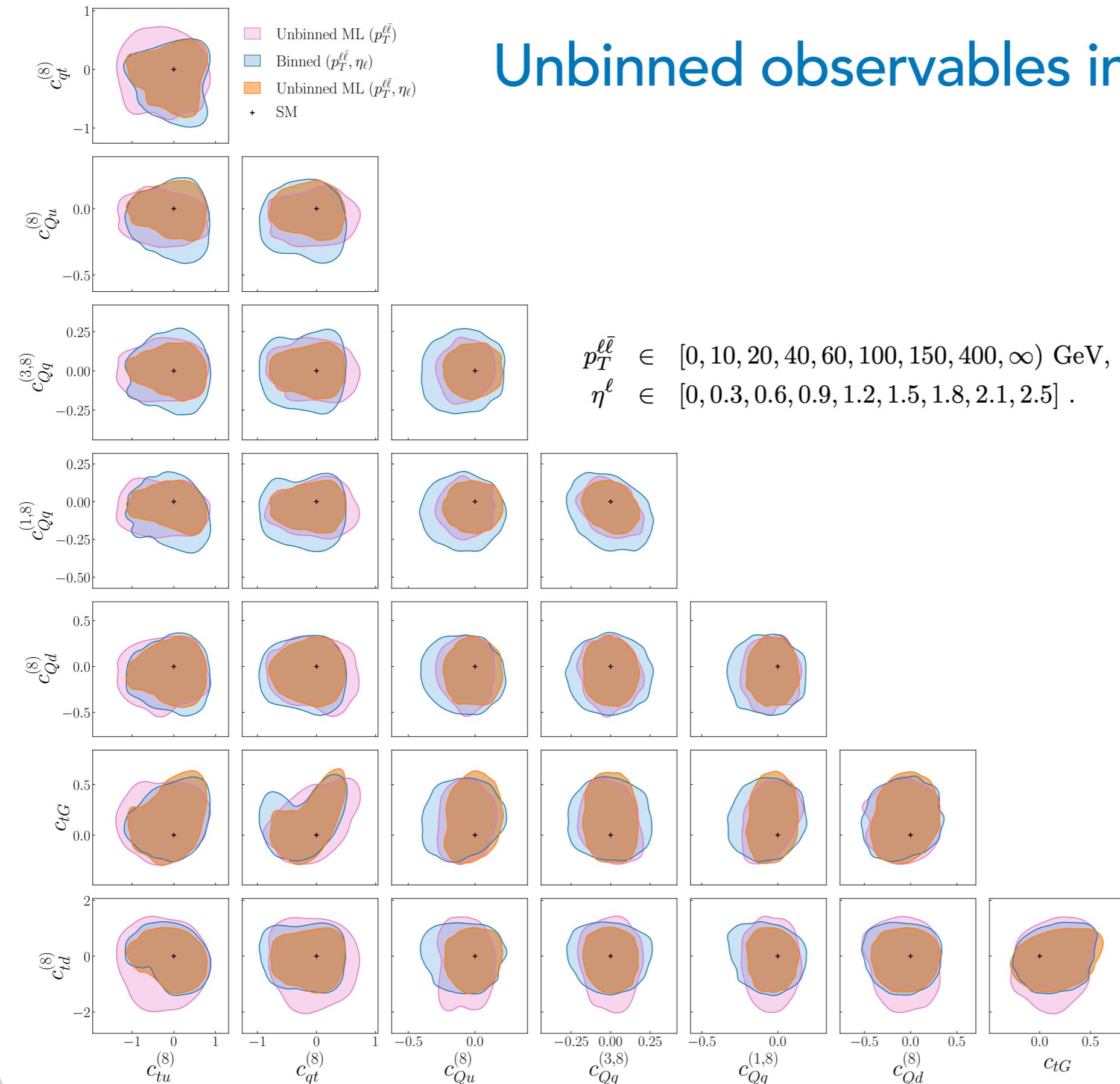
Unbinned observables in the top sector

Marginalised 95 % C.L. intervals, $\mathcal{O}(\Lambda^{-4})$ at $\mathcal{L} = 300 \text{ fb}^{-1}$



- ▶ **Binned vs unbinned** in $(p_T^{\ell\bar{\ell}}, \eta_\ell)$ small improvement relative to binned setup
- ▶ **2 features vs 18 features:** big increase in sensitivity

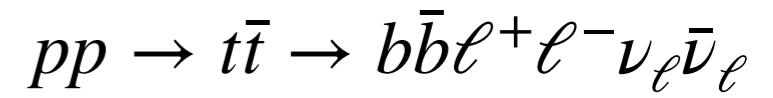
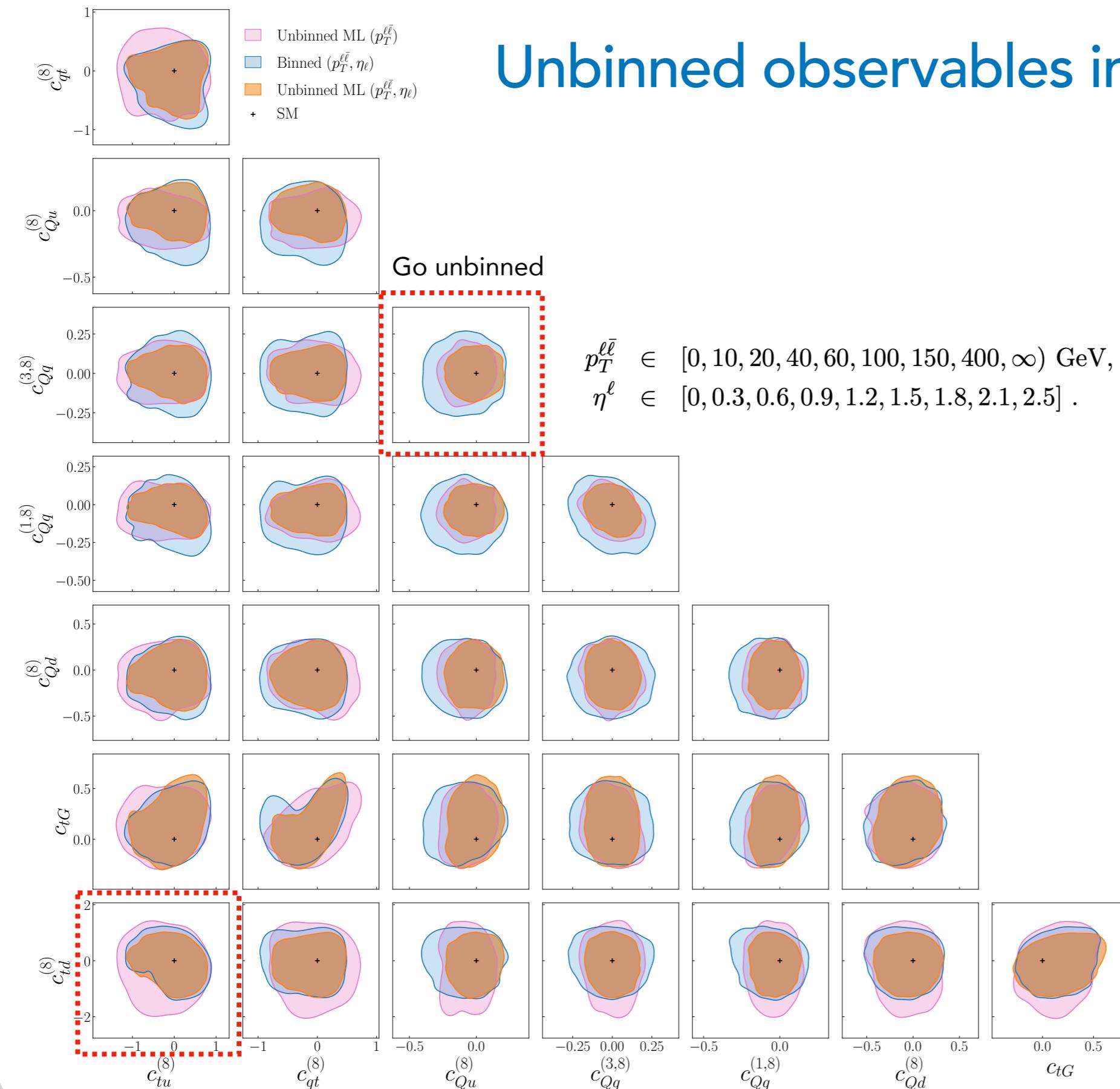
Unbinned observables in the top sector



$$\begin{aligned} p_T^{\ell\bar{\ell}} &\in [0, 10, 20, 40, 60, 100, 150, 400, \infty) \text{ GeV,} \\ \eta^\ell &\in [0, 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.5] . \end{aligned}$$

- Unbinned multivariate data is advantageous to constrain the EFT parameter space!
- Adding extra features or going unbinned?

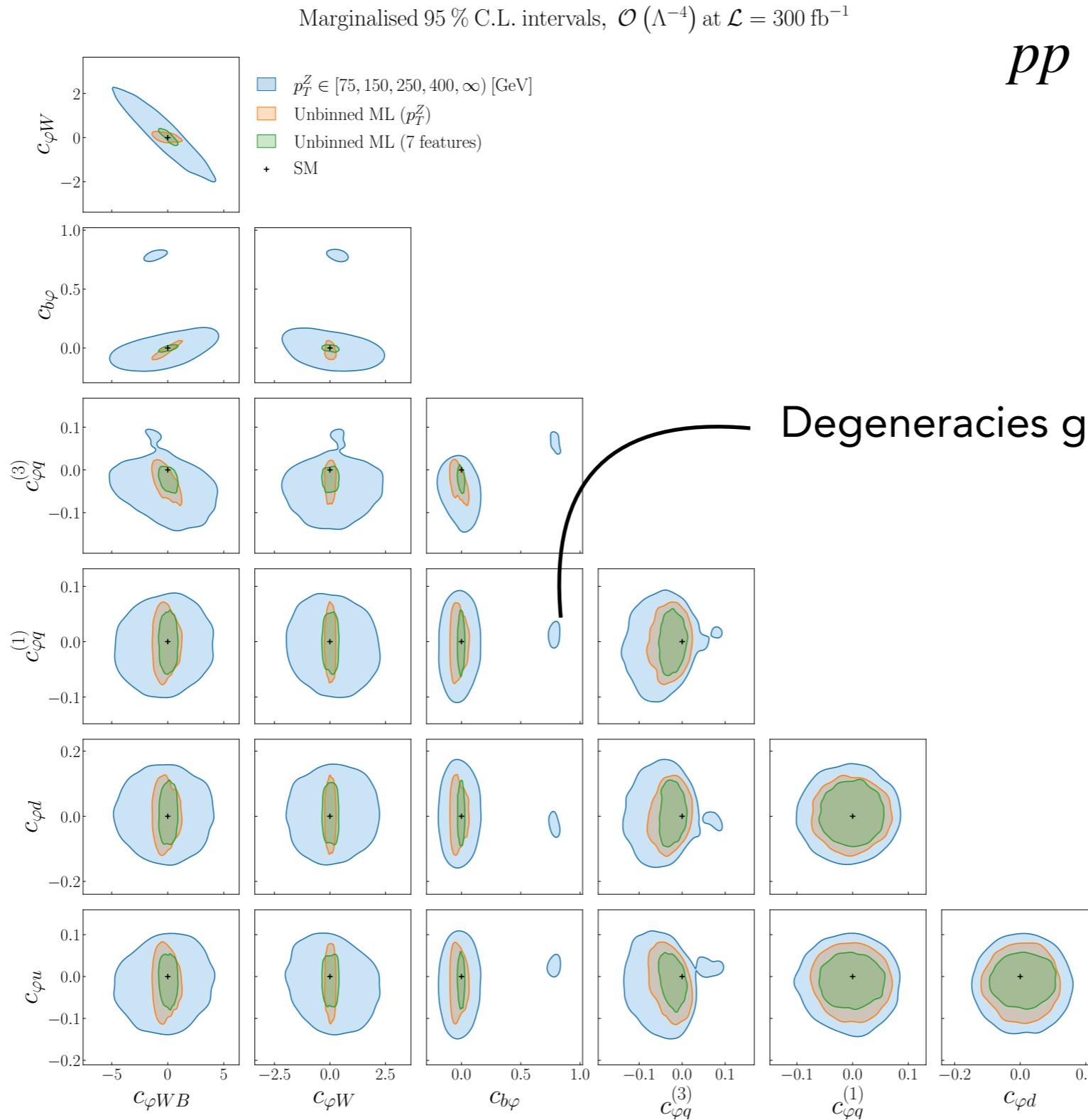
Unbinned observables in the top sector



$$\begin{aligned} p_T^{\ell\bar{\ell}} &\in [0, 10, 20, 40, 60, 100, 150, 400, \infty) \text{ GeV,} \\ \eta^\ell &\in [0, 0.3, 0.6, 0.9, 1.2, 1.5, 1.8, 2.1, 2.5] . \end{aligned}$$

- Unbinned multivariate data is advantageous to constrain the EFT parameter space!
- Adding extra features or going unbinned?

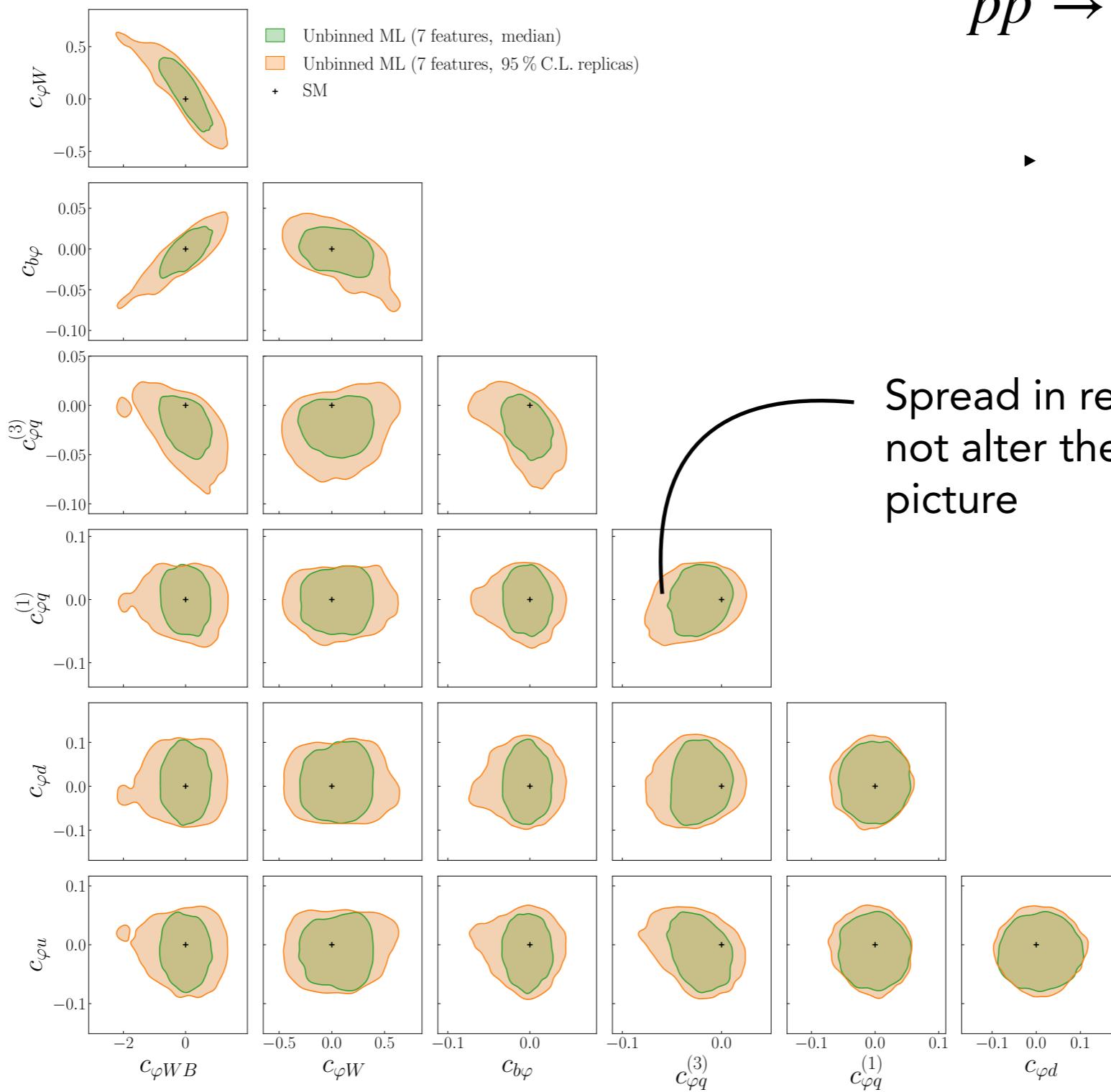
Unbinned observables in Higgs + Z associated production



Unbinned observables in Higgs + Z associated production

Marginalised 95 % C.L. intervals, $\mathcal{O}(\Lambda^{-4})$ at $\mathcal{L} = 300 \text{ fb}^{-1}$

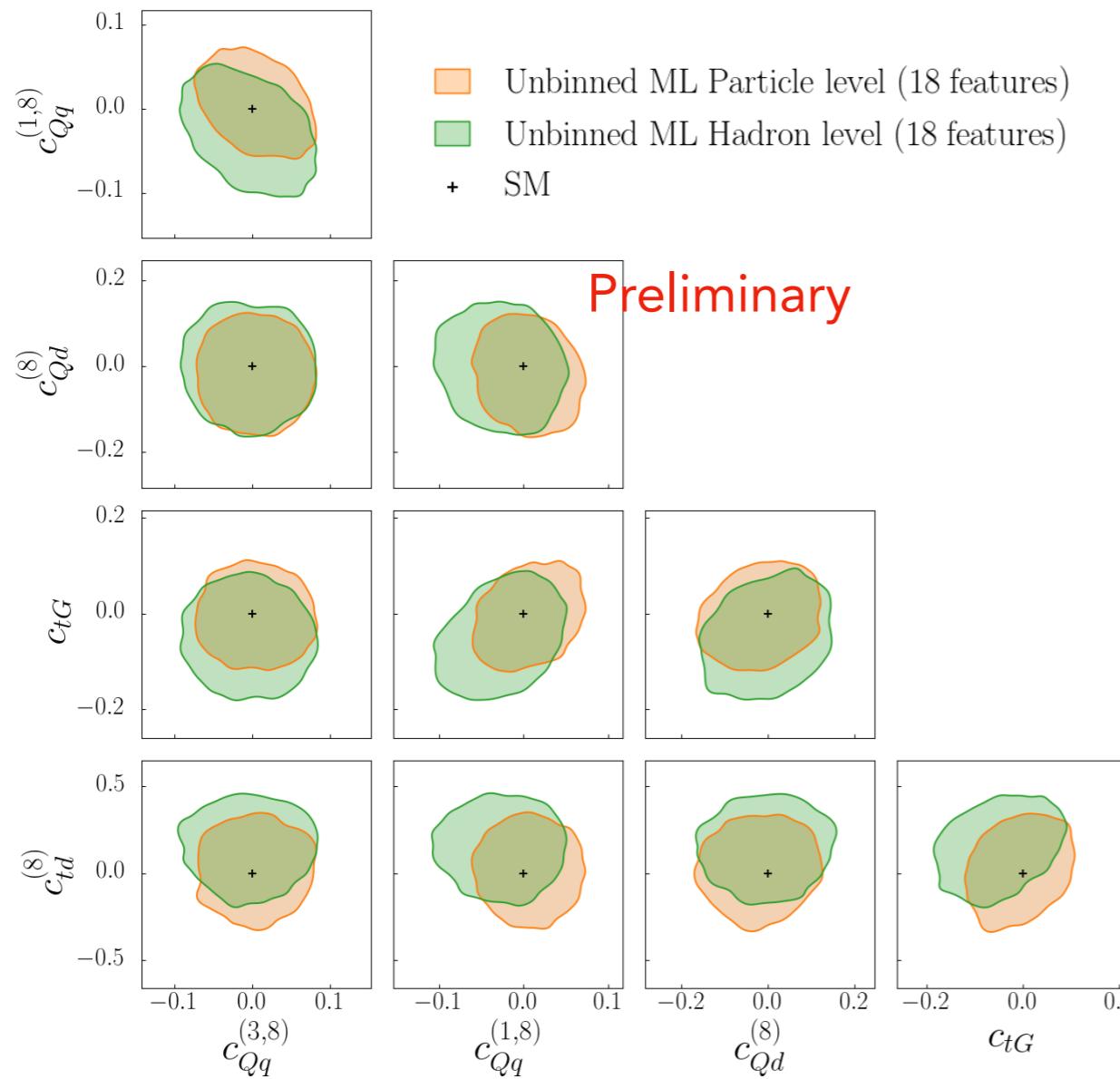
$$pp \rightarrow hZ \rightarrow b\bar{b}\ell^+\ell^-$$



Ongoing efforts

1. Hadronised level

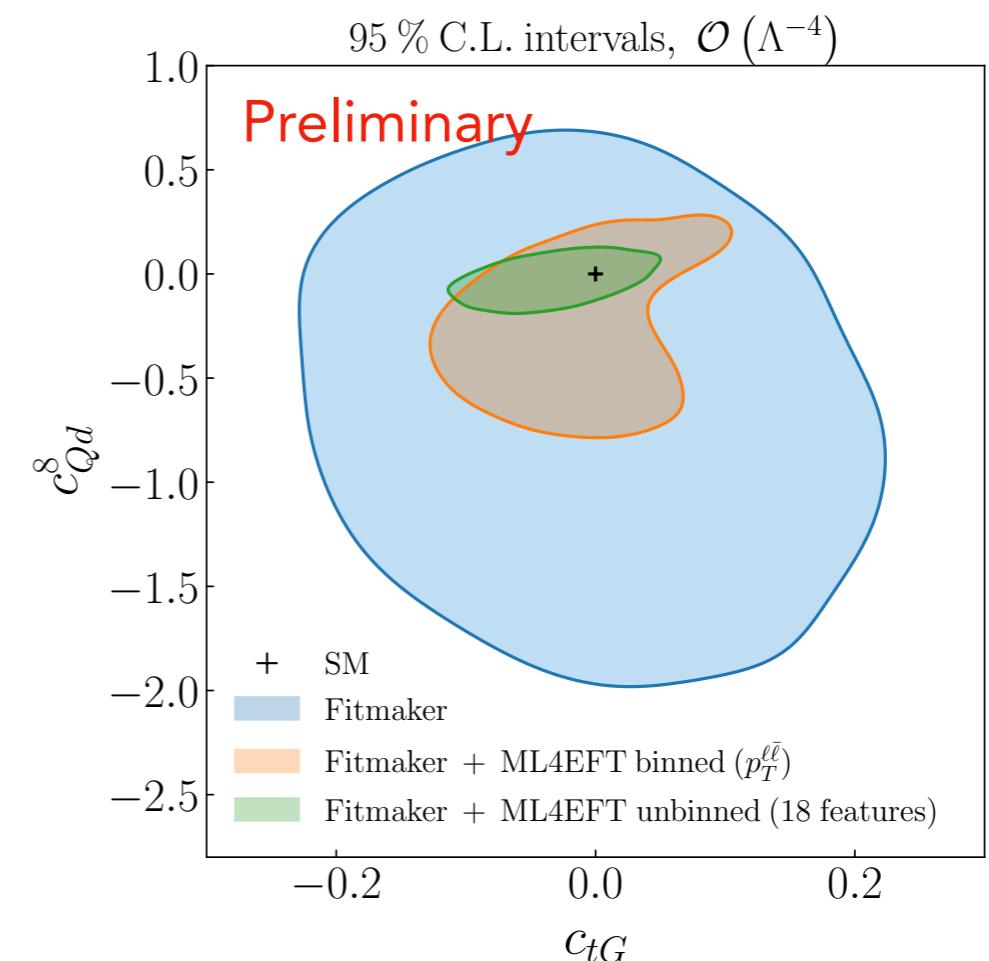
Marginalised 95 % C.L. intervals, $\mathcal{O}(\Lambda^{-4})$ at $\mathcal{L} = 300 \text{ fb}^{-1}$



MSc project by Pim Herbschleb

2. Integration into global fits

$$\log\mathcal{L}(c) = \sum_{k=1}^{N_D^{\text{(unbinned)}}} \log\mathcal{L}_k^{\text{unbinned}}(c) + \sum_{k=1}^{N_D^{\text{(binned)}}} \log\mathcal{L}_k^{\text{binned}}(c)$$



Conclusion and outlook

- Global EFT fits based on unbinned observables **enhance** the sensitivity significantly
- **ML4EFT** has efficient scaling properties, as required for global EFT fits and accounts for methodological uncertainties
- WIP: Integration into existing global fit frameworks
- WIP: benchmark study with A. Glioti et al.
- Please visit **ML4EFT** on GitHub (documentation + **tutorial**)

lhcfitnikhef.github.io/ML4EFT

Conclusion and outlook

- Global EFT fits based on unbinned observables **enhance** the sensitivity significantly
- **ML4EFT** has efficient scaling properties, as required for global EFT fits and accounts for methodological uncertainties
- WIP: Integration into existing global fit frameworks
- WIP: benchmark study with A. Glioti et al.
- Please visit **ML4EFT** on GitHub (documentation + **tutorial**)

lhcfitnikhef.github.io/ML4EFT

Thank you!